ON THE SYMBOLIC REGRESSION AND LOGISTIC EQUATION CONTROL

[1]LUKAS TOMASZEK, [2]IVAN ZELINKA, [3]MOHAMMED CHADLI

VSB-TU Ostrava, Czech Republic, University of Picardie Jules Verne Amiens, France
E-mail: [1]lukas.tomaszek@vsb.cz, [2]ivan.zelinka@vsb.cz, [3]mchadli@u-picardie.fr

Abstract - This article deals with the control equation synthesis by means of symbolic regression. Namely, the analytic programming and self-organizing migrating algorithm were used for a feedback equation creation and as a system the Logistic equation was used. Obtained equations are reported here and they are compared, by means of generalization for the different initial conditions.

Index Terms - Self-Organizing Migrating Algorithm, Symbolic Regression, Logistic Equation, Control.

I. INTRODUCTION

Self-organizing migrating algorithm (SOMA) is a swarm algorithm created by Zelinka [1]. Since the first presentation, the algorithm has been improved several times and has been used for solving real-world problems like an image enhancement [2], an automated waveform tuning [3] or a brain tissue microstructure measure [4]. In this article, we want to use the SOMA in a control domain. By control, we mean a process, where a system is controlled to the desired behavior. For example, one periodic behavioral like on Fig. 1, or two periodic behavioral like on Fig. 2.

From classical control methods, we can name time-delay auto synchronization (TDAS) [5], which can be expressed by equation

$$F_n = K \cdot (x_{n-m} - x_n) \quad (1)$$

or extended time delay auto synchronization (ETDAS) [6], which can be expressed by equations

$$F_n = K \cdot \left((1-R) \cdot S_{n-m} - x_n\right), S_n = x_n + R \cdot S_{n-m}, \quad (2)$$

where K and R are constants and $S_n \equiv \sum R^{a-1} x_{n-a \cdot m}$.

Not only classical methods, but unconventional methods like evolutionary algorithms [7] may be used in control. For example, Senkerik, in [8], [9], uses the SOMA for estimation the constants for the TDAS and the ETDAS. In [10], authors use the SOMA for identification of chaotic systems. Another example may be found in [9], [11], where authors use the SOMA and analytic programming (AP) [12] for chaos synthesis.

The AP [12] is a symbolic regression method, which allows us to synthesize complex programs from simple elements. For example, we can synthesize complex equation from basic mathematical operations. In this article, we want to evaluate whether the AP may be used for a feedback control equation creation and how the synthesize equation control the system for random initial conditions.

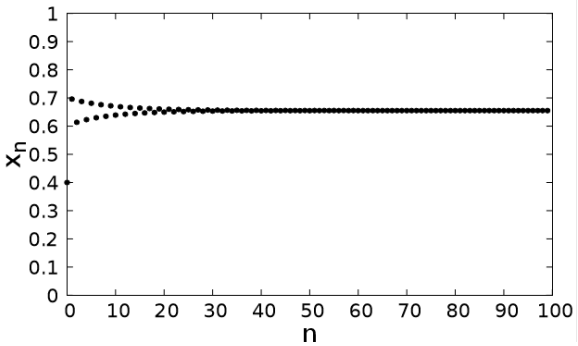


Figure 1: Periodic behavior with period equals to 1 (P1 orbit)

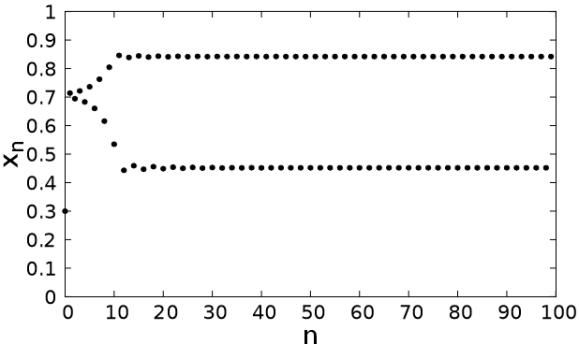


Figure 2: Periodic behavior with period equals to 2 (P2 orbit)

II. SELF-ORGANIZING MIGRATING ALGORITHM

SOMA is a swarm optimization algorithm based on the social behavior of competitive-cooperating individuals [13, 14]. Before the start of the algorithm, we have to set up the parameters, and we have to define a cost function. We can see all the required parameters, also with the recommended range, in Table I.

The run of the algorithm starts with an initial population creation. This population is randomly generated and distributed over searching space. After that, we start migration loops. In each migration loop, we firstly select a leader. The leader is the best individual (the individual with the best cost value).

After the leader is selected, each individual jumps towards him according to the equation

Table I: SOMA parameters

Parameter name	Recommender range
pathLength	[1.1, 5]
step	[0.11, pathLength]
PRT	[0, 1]
dimension	Given by a problem
popSize	[0, up to user]
migrations	[10, up to user]

$$x_{i,j}^{ML+1} = x_{i,j,start}^{ML} + \left(x_{L,j}^{ML} - x_{i,j,start}^{ML}\right) \cdot t \cdot PRTVector_j. \quad (3)$$

$x_{i,j}^{ML+1}$ is the value of i-th individual's j-th parameter in step t and migration loop ML + 1. $x_{i,j,start}^{ML}$ is the value of i-th individual's j-th parameter staring position in actual migration loop. $x_{L,j}^{ML}$ is the value of leader's j-th parameter in actual migration loop. t is a step from 0 by step to pathLength. PRTVector is a random vector of ones and zeros depended on PRT. If a randomly generated number is less than PRT, then in PRTVector will be 1 otherwise 0.

Each individual remembers all positions and the cost function values reached on these positions. After all jumps, the individual returns to the best position. Migration loops are repeated until we reach a maximum number of migration loops.

Above described SOMA is called SOMA with AllToOne strategy. This is not the only SOMA strategy. As an example of other strategies we can name:

- AllToOneAdaptive - In this strategy, the leader is replaced immediately, when we improve him, so it is replaced during the migration loop and not after it.
- AllToOneRand - In this strategy, the leader is not the best individual, but for each migration and each jumping individual, the leader is generated randomly.
- AllToAll - In this last mentioned strategy, each individual jumps to all other individuals in each migration loop.

For more details about the algorithm, please look at [1, 13].

III. ANALYTIC PROGRAMMING

Symbolic regression represented by AP [12] allows us to create complex programs, equations or units from simple elements. The AP consists of three basic parts - general function set (GFS), discrete set handling (DSH), and security procedures which allows us to avoid improper equation creation.

A. General function set

The AP has several GFSs based on the arguments number required for the given element. GFS_0 contains all elements - terminals which require zero arguments. In this set, we can find templates, inputs or constants like π. Next, GFS_1 contains all elements from GFS_0 and also functions which requires one argument like $\sin(x)$ or $\cos(x)$. Each GFS contains all elements from previous GFS plus new ones. The biggest set, marked as GFS_{all}, contains all elements from all sets.

B. Discrete set handling

Most evolutionary or swarm algorithms work with the real numbers, the main task of DSH is to convert the real number into the integer number. The easiest way how to do it is by rounding the real number. When the integer numbers are prepared, the equation can be created. DSH causes that the individual with real values is converted into a sequence of integers and from the sequence the equation is created.

Integer numbers serve as a pointer to the GFSs. Suppose we have $GFS_{ALL} = GFS_2 = \{+, -, \pi, e\}, GFS_0 = \{\pi, e\}$. Then, the integer sequence $1,3,4$ creates an equation $\pi + e$. The first number (1) picks from the GFS element is plus. Plus requires 2 arguments. Next two integer numbers are used for the argument creation. The first argument is given by the second number (3), so the first argument is π. The second argument is given by the third number (4), so the second argument is e. No other arguments are required, so the synthesis process ends with the final equation.

Constants are a special case. They can be stored in GFS_0 or GFS_1. In the first case, when they are stored in GFS_0, the addition metaheuristic algorithm needs to be performed, for suitable constant finding. In the second case, when they are stored in GFS_1, the constant needs an argument, and this argument can be taken directly from the sequence (unconverted real number).

C. Security procedures

Individuals have limited dimension. To be able to create equations which make sense, the security procedures need to be added. For example, if we use the GFS from the previous subsection, the sequence $1,1,1$ will create equation +++ and 4 additional arguments are required, but the sequence end, so we created an incomplete equation.

As an option, how to always create a correct equation, is to count the remaining numbers in the sequence and required number of arguments. After, when there is a possibility that the incomplete equation may be created, we picked the next element from the lower GFS. For example, if in the sequence remains only four numbers, and we need four arguments, all elements will be selected from GFS_0.

D.Summary

In summary, if we want to use the AP, to create an equation, we firstly have to define the GFSs and the security procedures. Also, we have to select the algorithm, on which the AP will be performed. Any optimization algorithm may be used, e.g. SOMA, PSO or differential evolution [15]. When GFSs and procedures are defined and the algorithm is selected, the AP may begin. Each algorithm individual is converted into an equation using DSH and mechanism described in subsection describing DSH. Each created equation is evaluated by a cost function, which defines how close we are to the required equation. In our case, the cost function defines how successful the control was. In this way, the algorithm evaluates individuals and approaches the optimal solution.

IV. LOGISTIC EQUATION

Logistic equation is a simple model of population growth. A simple logistic equation may be expressed by equation

$$x_{n+1} = r \cdot x_n \cdot (1 - x_n). \quad (4)$$

Logistic equation is strongly dependent on parameter r, and with r > 3.57 becomes chaotic.

V. EXPERIMENT DESIGN

In this section, we firstly describe used parameters and setup for the experiments and after in detail, we describe the whole experiments.

A. Parameters and setup

In the experimental part, we use AP for feedback equation synthesis. This equation should control the logistic equation, with $r = 3.8$, to one periodic behavioral. Our GFS contains operations plus, minus, divide and multiply; functions sinus and cosinus; variables x_n and x_{n-1}; and constant.

Some security procedures need to be added to the synthesized equation, to ensure that the values x_n are in range $[0,1]$. To ensure that, the inputs are from interval $[-F_{MAX}, F_{MAX}]$, where we set $F_{MAX} = 0.02$. If the counted value is greater than F_{MAX}, F_{MAX} is used. As an algorithm which uses the AP we chose the SOMA. The algorithm parameters were set according to Table II. The cost function was defined according to equation

$$CF = \sum ((x_n - x_{n-1})^2 \cdot n^2) \quad (6)$$

where x_n is a state of the logistic equation. The lower the difference between the x_n and $x_{n+1}$is, closer we are to the required one periodical orbit behavioral. n^2 causes that the stabilization to one periodical orbit in later iterations will be penalized.

Table II: Used SOMA parameters

Parameter name	Recommender range
pathLength	3
step	0.3
PRT	0.3
dimension	5, 10, 20
popSize	20
migrations	100

B. Experiments detail

In the experimental part, SOMA, AP and Logistic equation were used with the parameters mentioned in the previous section. Each new SOMA individual was converted using AP to a feedback control equation F_x. This equation was used to control the Logistic equation from the same initial value x_0, so the Logistic equation runs from the same initial condition for each individual. Controlled Logistic equation looks like in equation

$$x_{n+1} = r \cdot x_n \cdot (1 - x_n) + F_x. \quad (7)$$

A hundred iterations of Logistic equation were done and the error of individual given by a cost function was calculated and assigned to a given individual. After all individuals in all migration loops were evaluated, the best individual was selected and the equation created by this individual was captured. Final equation was adjusted by basic mathematical operations to make the simplest form. After, the given equation was used for Logistic equation control with different initial conditions. The states of the Logistic equation were captured and reported in the results section.

VI. RESULTS

In the following part, the results are reported. Six different obtained equations (8) – (13) with minimal errors are reported here.

$$F_x = (x_n - x_{n-1}) \cdot (x_{n-1} \cdot x_n) \quad (8)$$

$$F_x = (x_{n-1} - x_n) \cdot (k - x_{n-1}); k = 0.1685109523 \quad (9)$$

$$F_x = k \cdot (x_{n-1} - x_n) \cdot x_{n-1}; k = -0.7057668905 \quad (10)$$

$$F_x = k \cdot (x_n - x_{n-1}); k = 0.6304772353 \quad (11)$$

$$F_x = x_n - \frac{k}{x_n}; k = 0.5414329538 \quad (12)$$

$$F_x = x_n - k; k = 0.7453371604 \quad (13)$$

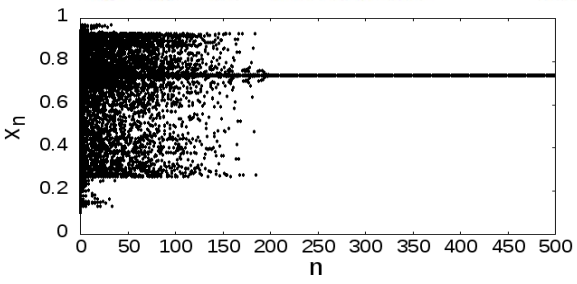


Figure 3: States of controlled Logistic equation for different initial values with control equation (8)

These equations were used for Logistic equation control, and the states for different initial conditions are reported as plots (Fig. 3 - Fig. 8).

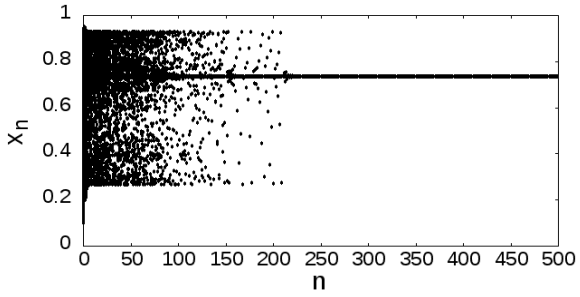


Figure 4: States of controlled Logistic equation for different initial values with control equation (9)

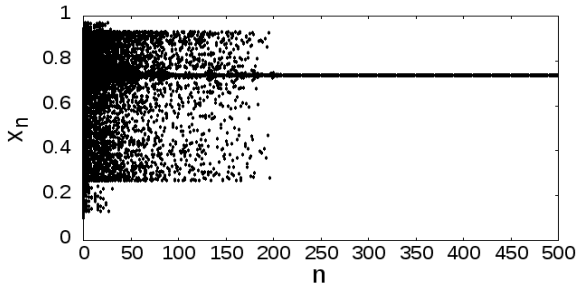


Figure 5: States of controlled Logistic equation for different initial values with control equation (10)

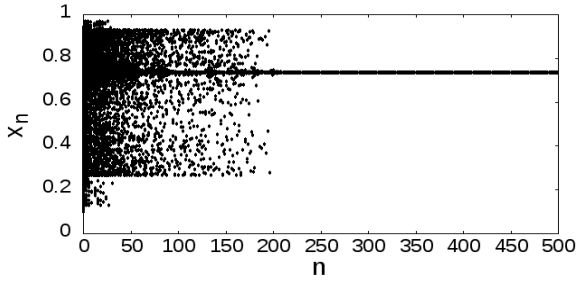


Figure 6: States of controlled Logistic equation for different initial values with control equation (11)

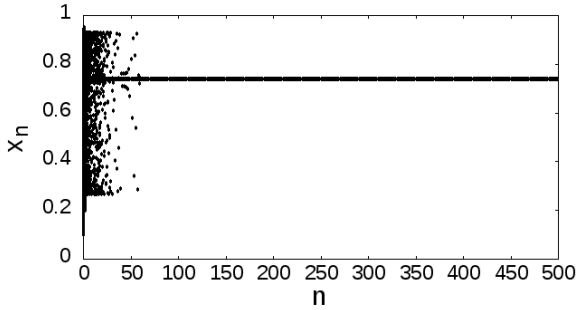


Figure 7: States of controlled Logistic equation for different initial values with control equation (12)

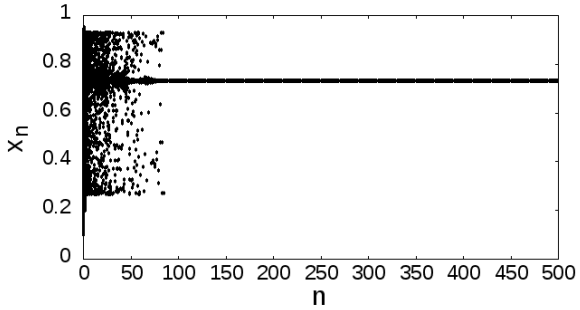


Figure 8: States of controlled Logistic equation for different initial values with control equation (13)

CONCLUSION

In this article, we showed that the AP is capable to synthesize a feedback control equation. We reported 6 obtained equations and showed how the behavior will look for different initial conditions. All obtained equations were capable to control the Logistic equation to the one periodical orbit behavioral. Some of them faster and some of them slower. The best result was obtained by the control equation (12).
We can also see, that one of the synthesize equation was the same as the TDAS control equation. This means, that AP was capable to synthesize already known equation for control, and also other equation, which shows similar behavioral.

ACKNOWLEDGEMENT

This work was supported by the ESF in 'Science without borders' project, reg. nr. CZ.02.2.69/0.0/0.0/16\ 027/0008463 within the Operational Programme Research, Development and Education, and by SGS SP2019/137 VSB-TU Ostrava, during the research internship at MIS-UPJV Amiens France.

REFERENCES

[1] Ivan Zelinka. Soma—self-organizing migrating algorithm. In New optimization techniques in engineering, pages 167–217. Springer, 2004.

[2] Anupriya Gogna and Akash Tayal. Comparative analysis of evolutionary algorithms for image enhancement. arXiv preprint arXiv:1312.5045, 2013.

[3] Lars Nolle, Ivan Zelinka, Adrian A Hopgood, and Alec Goodyear. Comparison of an self-organizing migration algorithm with simulated annealing and differential evolution for automated waveform tuning. Advances in Engineering Software, 36(10):645–653, 2005.

[4] Daniel C Alexander. A general framework for experiment design in diffusion mri and its application in measuring direct tissue-microstructure features. Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine, 60(2):439– 448, 2008.

[5] Kestutis Pyragas. Continuous control of chaos by self-controlling feedback. Physics letters A, 170(6):421–428, 1992.

[6] Kestutis Pyragas. Control of chaos via extended delay feedback. Physics letters A , 206(5-6):323–330, 1995.

[7] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. Evolutionary algorithms for solving multi-objective problems, volume 5. Springer, 2007.

[8] Roman Senkerik, Ivan Zelinka, and Eduard Navratil. Optimization of feedback control of chaos by evolutionary alghorithms. IFAC Proceedings Volumes, 39(8):77–82, 2006.

[9] Roman Senkerik, Ivan Zelinka, and Zuzana Oplatkova. Evolutionary blackbox control of logistic equation. In 2009 European Control Conference (ECC), pages 2652–2657. IEEE, 2009.

[10] Ivan Zelinka, Roman Senkerik, Zuzana Oplatkova, and Donald Davendra. Evolutionary identification of chaotic system. IFAC Proceedings Volumes, 42(7):308–315, 2009.

[11] Ivan Zelinka, Mohammed Chadli, Donald Davendra, Roman Senkerik, and Roman Jasek. An investigation on evolutionary reconstruction of continuous chaotic systems. Mathematical and Computer Modelling, 57(1-2):2–15, 2013.

[12] Ivan Zelinka, Zuzana Oplatkova, and Lars Nolle. Analytic programming–symbolic regression by means of arbitrary evolutionary algorithms. Int. J. of Simulation, Systems, Science and Technology, 6(9):44–56, 2005.
[13] Donald Davendra, Ivan Zelinka, et al. Self-organizing migrating algorithm. New Optimization Techniques in Engineering, 2016.
[14] Ivan Zelinka. Soma–self-organizing migrating algorithm. In Self-Organizing Migrating Algorithm, pages 3–49. Springer, 2016.
[15] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4):341–359, 1997.

★★★